CSCI 201L - Final Project
30.0% of course grade

Introduction

Throughout this course you have learned or will learn topics including user interface design, GUI components, multi-threaded programming, and networking. Up until this point in the semester, you have worked alone on homework assignments that have had strict guidelines. This project is going to expose you to working in a group on a large project of your own creation from conception to deployment. You will need to utilize many of the topics you have learned throughout the course, as well as some topics from outside the course, in this project.

Description

You will be working in groups of 4-6 people to create an application of your choosing. The groups will be randomly formed to more accurately mimic working with people at a company (where a programmer rarely gets to determine all of the people on the team). NOTE: *I surveyed the fall 2016 class to ask if they thought it was a good idea to have the teams randomly assigned, and 81.3% of the students (152/187 students) thought it was.*

You must abide by the specifications below, though anything not specified can be implemented however you want. You will develop a program with which users can interact. The software may cover any functional category (e.g., game, tool, travel, communication, etc.). Below are some requirements:

- The program has user login functionality and a central server to authenticate.
- Users can interact with the software as a guest but experience limited functionality compared to authenticated users.
    ◦ Users must be able to register so they can become an authenticated user. The method by which they register is up to you.
    ◦ When a user registers, the data should be stored in a database on the server.
- The program must be aesthetically-pleasing (i.e. make it look nice).
- The program must have multi-threaded and network functionality outside of chatting. You can include chat, but there must be multi-threaded and networking functionality other than that as well.
- At least the back-end server logic must be written in Java. It could be a web application, mobile app, or standalone application, but *at least* the back-end server code must be written in Java.

If you struggle to come up with an idea or want to know if an idea is an appropriate endeavor, feel free to discuss with your professor or one of the CPs. This description is intentionally vague

so that you can come up with whatever project you want within the above restrictions. Some project ideas from current and former CPs when they took 201 are posted on the Assignments page, so that may be a good place to start.

Group Work

Part of the purpose of this project is to work together in a team. There will most likely be some issues within the team, and I hope that you all will work out the issues. Each group will be assigned a CP with whom to work throughout the project. If you are unable to resolve issues within the group, please let your CP know. If the problem persists, let the professor know so that a discussion can be mediated among the team members. In extreme cases, groups can be dissolved or people removed from groups. At that point, the sub-groups at fault will be responsible for the entire application.

We will also be monitoring the progress of each student based on code committed to a Github repository through a custom tool we have developed. This will allow the course staff to monitor the number of files and number of lines committed on a continuous basis. Although this is not specifically included in your grade, this will alert the staff to any problems that may be occurring so we can follow up with the team or individual students.

During the group presentation each member will showcase his/her contribution to the project. If you let your group do all the work, you will have nothing to show, and this will affect your individual grade. There will be two peer reviews throughout the semester where your team members will rate you. This feedback should be taken constructively to improve the project, though your teammates' reviews will not affect your final grade directly.

Specific Requirements

You will need to abide by good object-oriented guidelines with your program. Part of your grade will be based on the efficiency of the design and implementation of your code as portrayed in your documentation as well as your code. You will meet weekly with your assigned CP during the project to discuss any issues and provide a status update.

With multiple programmers on a team, you will need to set up a version control system for managing your program. There are multiple version control systems available, including CVS, SVN, git, etc., and we are requiring use of Github. You will need to provide the course staff with access to your repository. A tutorial on how to do this is provided on the Assignments page of the course web site.

You will need to submit documents for each major phase in the software engineering lifecycle – concept, high-level requirements, technical specifications, detailed design document, testing document, implementation (code), and deployment document. Your final program will be demonstrated during a specified time, and you will show all of the features of your program. We will not run specific test cases during the demonstration, though we may ask you about specific functionality.

Tips

Since there are a lot of parts to the project, you may be tempted to assign specific parts to individual team members. This is fine, but it is important that everyone in the group reviews each part before submission. **You should probably have an internal deadline of at least one day earlier than the actual submission deadline so the rest of the group is able to review each document before submitting.** Of course, with the implementation, this should be done much earlier so the entire group is able to test and work out bugs.

Much of the work might be able to be accomplished separately, especially with the technologies we have at our disposal for remote work. I am still going to require at least two hours a week to meet as a group outside of class and work out any issues and discuss upcoming tasks. **Based on feedback from past groups, having a specific time each week allocated to the team physically working together in the same location has proven very beneficial.**

Submission Instructions

All of the documents will be submitted as PDFs via Blackboard. Only one person from the group needs to submit each document, but make sure all team members' names are on **every** document so we can assign points accordingly.

In Eclipse, your project should be named "CSCI201_FinalProject_groupname". Your group name should be something unique from other groups in the class and meaningful for your project. For example, if your project is Monopoly, your group name could be "monopoly", and you would name your project "CSCI201_FinalProject_monopoly". This will help us for grading in case we need to run your program after submission.

You should always have a comment at the top of each source code file with (at least) all of your team members' names in it. Submit the implementation of your final project as a zip file through Blackboard. You can export the code from Eclipse as a .zip.

% Final Grade	Criteria
4.0%	Weekly meetings with assigned CP (8 total meetings at 0.5% each)
1.0%	Submit team members with emails, weekly meeting time, and project proposal
2.0%	High-Level Requirements document
2.0%	Technical Specifications document
3.0%	Detailed design document
2.0%	Testing document
1.0%	Peer Review #1
1.0%	Deployment document
2.0%	Code Complete (though testing and bug fixes can still be implemented)
1.0%	Peer Review #2
1.0%	Complete documentation
10.0%	Demonstration/Presentation

Description of Grading Criteria
All documents should be submitted as PDFs.

Weekly meetings with assigned CP (8 total meetings, totaling 4.0%)
As soon as the project is explained in class, you will be assigned a team of students with which to work for the project. You will need to get in touch with your group members and your assigned CP to set up a two weekly meeting times. The meeting should take no more than 30 minutes, and you will provide your CP with an update on the status of the project and explain any problems or issues that have arisen. The weekly meetings are graded based on attendance and participation.

Total Grade	0.5%
0.25%	**Showing up to the assigned meeting on time (deduct 0.05% for every 5 minutes a student is late)**
0.25%	**Participating in the meeting**

Peer Reviews (2 during the project, totaling 2%)

The peer reviews will be submitted by each member of the team with a review about all of the other people on the team. The peer review will be returned to the other members of the team in an aggregated and anonymous fashion. The URL of the peer reviews will be provided to you before the due date. The points on the peer review will be given based on whether you completed the review for all of the members of your team.

Total Grade	1.0%
1.0%	**Submitting peer review for all of your teammates**

Submit team members, weekly meeting time, and project proposal (1%)

You will need to provide the names and email addresses of your 4-6 member group. You also need to find a two hour block where the entire team is able to meet each week. Determine a location to meet. You must also come up with the idea for your project. Submit the idea along with a brief overview of the project which highlights how you will use a GUI, multi-threaded code, and networking. This proposal should be less than one page in length and should only provide the concept of the project. The specifics of the implementation and other parts of the project will come in future documents.

Total Grade	1.0%
0.5%	**Document includes team member names, email addresses, weekly meeting time, and 1-2 paragraph project proposal**
0.5%	**Formatted professionally with no grammatical mistakes**

High-Level Requirements document (2%)

All teams will need to write up all of the details about their project. You must write out these requirements as if you were a client requesting for a program to be written that performed the tasks provided. The document must be an overall description of the behavior of your program.

Total Grade	2.0%
1.0%	**All requirements for project are included without technical details**
0.5%	**Formatted professionally**
0.5%	**No grammatical mistakes**

Technical Specifications document (2%)

Based on the feedback you receive on your requirements document, you will then need to write up technical specifications. You will submit both your requirements document and your technical specifications document since you may have had to modify your requirements based on provided feedback. Make sure you clearly delineate anything that you added, deleted, or changed in your specifications. This can be accomplished using Track Changes in Microsoft Word (or a similar feature in other word processors).

Total Grade	2.0%
0.5%	**All tasks from High-Level Requirements included**
0.25%	**Tasks include hours that do not exceed 8 hours**
0.5%	**Tasks contain descriptions that are sufficient for design phase even if designer was unfamiliar with application**
0.5%	**Formatted professionally**
0.25%	**No grammatical mistakes**

Detailed Design document (3%)

Based on the feedback you receive on your technical specifications document, you will write up your detailed design document. This should show any difficult algorithms, the exact GUI (though slight modifications will still be allowed), the database schema (ER diagram), the hardware/software requirements, class diagrams, inheritance hierarchies, etc. This will likely be a lengthy document.

In addition, submit your requirements and specification documents again with any changes clearly delineated using a feature such as Track Changes.

Total Grade	3.0%
1.0%	**All of the specifications in the Technical Specifications document are designed out**
0.25%	**GUI mockup is provided in some manner**
0.25%	**Database schema provided**
0.25%	**Hardware and software requirements provided**
0.25%	**Class diagram and/or inheritance hierarchy provided**
0.5%	**Document is formatted professionally**
0.5%	**Document has no grammatical mistakes**

Testing document and testing code (2%)

You will need to prove to us that your program is working the way it is supposed to. That means that you will need to include sufficient test cases to prove that your program is working. This includes all types of testing, including white box testing, black box testing, unit testing, regression testing, etc. Not only should you create a document in this step, but you should also create the test code. You will be able to add other test cases and code throughout the implementation phase, but this will hopefully provide you with a good set of test cases to ensure proper operation of your application.

In addition to the testing document, submit your requirements, specifications, and detailed design with any changes clearly delineated using a feature such as Track Changes.

Total Grade	2.0%
1.0%	**Test cases that cover all of the functionality in the program are included, with inputs and expected outputs**
0.5%	**Document is professionally formatted**
0.5%	**Document has no grammatical mistakes**

Code Complete (though testing and bug fixes can still be implemented, 2%)

You will submit your code wrapped up in a zip file that can be imported into Eclipse and executed. By this date, the new development of your program should be completed, and only bug fixes and testing should be ongoing. You shouldn't be adding new features into your program after this date.

In addition to the code, submit your requirements, specifications, detailed design document, and testing document/code with any changes clearly delineated using a feature such as Track Changes. You probably will have added more test cases as development proceeds, so mark those.

When you submit your implementation, include a `README.txt` file in your project in the `doc/` folder of your project that explains what works and what doesn't, along with any other nuances with your project about which the graders should be aware.

Total Grade	2.0%
2.0%	**Final project code submitted as a zip file**

Deployment document (1%)

We won't actually be deploying your program in a live environment, but if we were to, this document should be what is followed. It needs to include step-by-step instructions of what to do to take the code from Eclipse and deploy it. This will most likely consist of some other configurations also (i.e. outside libraries, server settings, Java download/install, etc.).

In addition to the deployment document, submit your requirements, specifications, detailed design document, testing document/code, and updated code with any changes clearly delineated using a feature such as Track Changes.

Total Grade	1.0%
0.5%	**Step-by-step instructions seem complete for getting the application running in a different environment**
0.5%	**Formatted professionally with no grammatical mistakes**

Demonstration/Presentation (9%)

You will demo your program and prove to us the correctness of it. We may ask you to show something specific, but you will be driving the demonstration. Also during this time, each group member will have a chance to showcase what he/she contributed to the project. You will have to walk us through the program and show us all of the features. I assume this will take between 5-10 minutes.

There are additional items that need to be discussed during the final presentation. This will hopefully demonstrate to you and the class that you have an understanding of some of the soft skills that are needed to complete a project. The specific topics are included in the grading rubric posted on the Assignments page of the course web site.

The slides used during the presentation will need to be submitted to Blackboard.

The grade for the presentation is based solely on the professor's evaluation.

The grading rubric for the Demonstration/Presentation is provided on the Assignments page of the course web site.

Presentation Review (1%)

You will need to provide a review of presentations on the demonstration day.

Complete documentation (1%)

You will put all of your documents together into one document and include a cover page, table of contents, and page numbers. This entire document will then be submitted to Blackboard. Again, if anything has changed from the last submission, mark the changes clearly with a feature such as Track Changes. This document should be nice enough that it could be printed and presented to the client of this application as the final document.

Total Grade	1.0%
0.5%	**All documents are in a single PDF**
0.5%	**Cover page and table of contents included**